


يشمل جميع أنواع الكتابة من الانشاء والتقرير والمقال والرسالة وغيرها، وسنكتفي بكلمة بحث أو رسالة هنا

Simon Peyton Jones

Microsoft Research, Cambridge

ترجمة د. سوزان الخضير

docsu1970@gmail.com

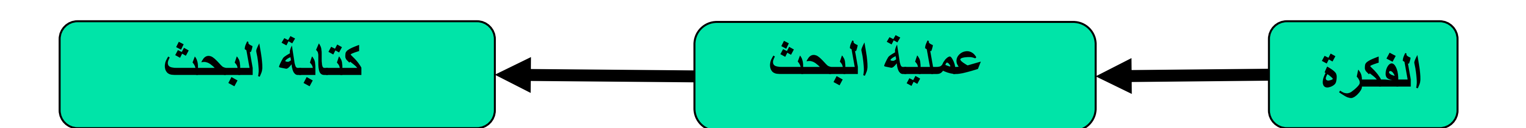
Twitter: @docsu1970

# كتابة البحث مهارة

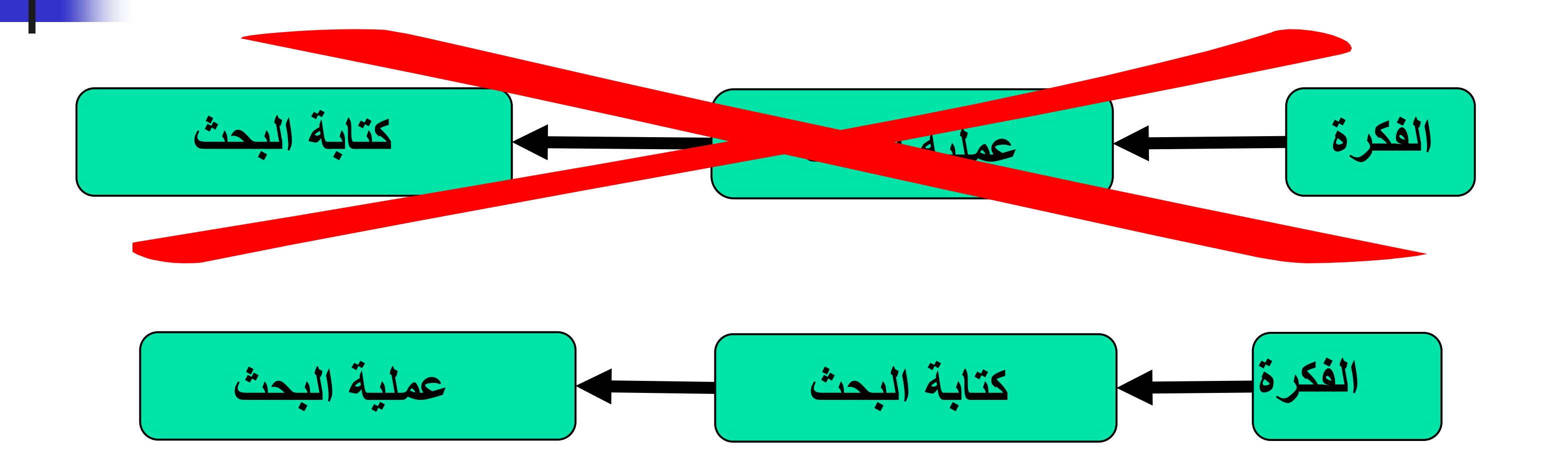
- كثير من البحوث مكتوبة رديئة الأسلوب والمحتوى
- الكتابة الجيدة (الصياغة أو التأليف) مهارة بامكانك تعلمها
- وهي مهارة تستحق أن تتعلمها:
  - ستجد قبولا أكثر لكتاباتك
  - سيكون لأفكارك تأثير أقوى
  - وستكون لديك أفكار أفضل

زيادة درجة الأهمية

# كتابة البحث: نموذج 1



# كتابة البحث: نموذج 2




- هذه الطريقة ترغمنا على التركيز والوضوح
- يحدد لنا بوضوح ما الذي لا نفهمه
- يفتح المجال للنقاش مع الآخرين: التأكد من الحقائق، النقد ، التعاون

# لا تشعر بالقلق تجاه فكرة كتابة بحث

**فكرة خاطئة** يجب أن تكون لديك فكرة عبقرية حتى تكتب بحثاً. (أليس الجميع كذلك؟)

**أكتب بحثا، أو الق محاضرة**

**عن**

# أي فكرة

**مهما بدت ضعيفة أو عديمة الأهمية في نظرك**

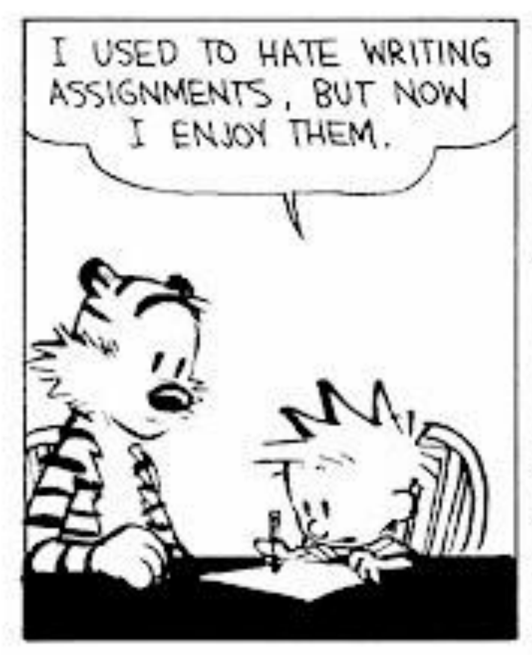
# لا تشعر بالقلق تجاه فكرة كتابة بحث

**أكتب بحثا، أو الق محاضرة عن أي فكرة، مهما بدت ضعيفة أو عديمة الأهمية في نظرك**

- **كتابة البحث هي طريقة تَكوُّن الفكرة لديك في الأساس**
- غالباً ما يظهر لك أن الموضوع أكثر امتاعاً وتحدّياً مما تصورت في البداية

# الغرض من البحث

# لماذا نُتعب أنفسنا؟










**فكرة خاطئة**

نحن نكتب أساساً من أجل أن نتميز في نظر الآخرين، وحتى نُعرف، ولننال الترقيات

# الأبحاث تنقل الأفكار

- هدفك: هو أن تُعدي ذهن قارئك **بفكرتك** كما يفعل الفيروس
- البحوث (الكتابة عموما) أكثر مقاومة للزمن من برامج الكمبيوتر (انظر الى موزارت)

**أعظم الأفكارعديمة القيمة لو انك احتفظت بها لنفسك**



# الفكرة



**الفكرة**

معلومة يمكن اعادة استعمالها، مفيدة للقارئ

- قررما هي فكرتك
- تأكد من أن فكرتك واضحة تماما لقترئ. كن واضحا 100%:
  - «...الفكرة الأساسية لهذه الرسالة هي...»
  - «... في هذا الجزء نقدم الأفكار التي تساهم بها الرسالة...»
  - الكثير من الرسائل تحتوي على أفكار جيدة، لكنها لا تقدمها بوضوح للقارئ

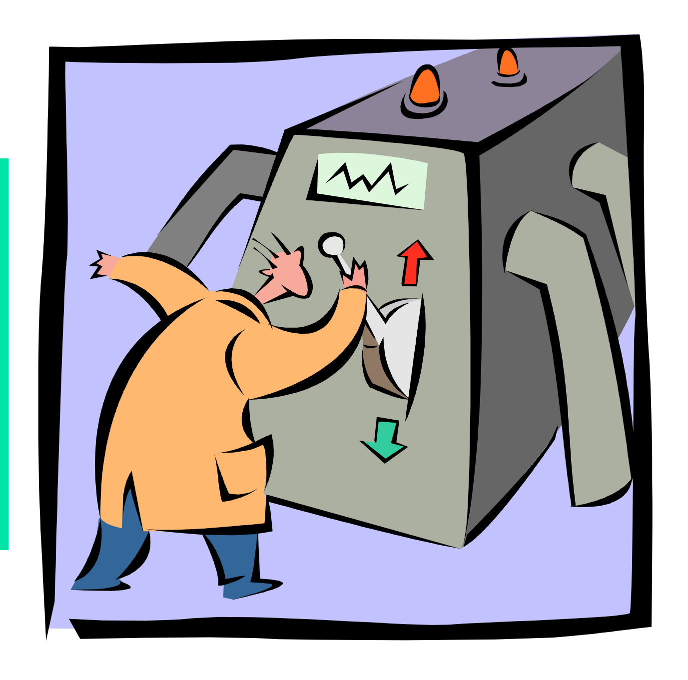
# رنّة واحدة

- تريد من بحثك أن يكون له رنّة واحدة (كرنّة الجرس) : رنّة (أو فكرة) واحدة واضحة وحادّة
- أعد قراءة بحثك، هل سمعت الرّنّة؟
- قد لا تعرف ما هي الرنة (الفكرة) تماما حين تبدأ الكتابة. لكن يجب أن تعرفها حين تنتهي
- اذا كانت لديك أفكار كثيرة فاكتب رسائل (بحوث) كثيرة

أشكر السيد جو توش لفكرة «رنة واحدة»

# ليس الغرض من رسالتك

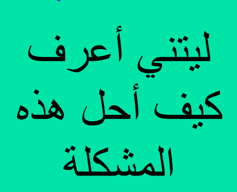
## شرح جهاز النظام الخزعبلي

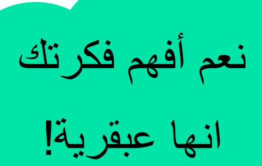


- القارئ لا يمتلك جهاز النظام الخزعبلي
- ولكنه يريد أفكارا قابلة للاستعمال مرارا، يستطيع هو أن يستفيد منها، وليس شرحا لجهاز أو نظام محدد الفائدة

# سياق الكلام

- اليكم هذه المشكلة
- انها مشكلة مثيرة للاهتمام
- وهي مشكلة لم تجد حلا بعد
- **هذه هي فكرتي (الفرضية)**
- كيف تفيد فكرتي (تفاصيل، حقائق)
- وهكذا تتقارن فكرتي مع أفكار ورؤى الآخرين






# هيكل البحث (رسالة مؤتمر)

- العنوان (1000 قارئ)
- الملخّص (أربع جُمل، 100 قارئ)
- المقدمة (صفحة واحدة، 100 قارئ)
- المشكلة (صفحة واحدة، 10 قرّاء)
- فكرتي (صفحتين، 10 قرّاء)
- التفاصيل (5 صفحات، 3 قرّاء)
- جهود سابقة (1-2 صفحة، 10 قرّاء)
- الاستنتاج والجهود المستقبلية (نصف صفحة)

# الملخص

- عادة أكتب الملخص في الآخر
- يعتمد عليه أعضاء اللجنه لاختيار الرسائل التي سيقرؤوها
- أربع جمل «حسب Kent Beck»

1. اذكر المشكلة
2. سبب اهمية المشكلة
3. مالذي ينجزه الحل الذي تقدمه
4. ما الخطوة التالية بعد حلك؟

# مثال

1. الكثير من البحوث رديئة الكتابة وصعبة الفهم
2. وهذا مؤسف لأن أفكارهم الجيدة لن تلقى اهتماما وتقديرا
3. اتباعك لخطوات ارشادية بسيطة (لكتابة البحث) بامكانه تحسين مستوى رسالتك
4. سيُستفاد من عملك مرات أكثر وآراء الآخرين ونقدهم الذي يصلك سيكون له دور في تحسين أبحاثك

# الهيكل

- الملخص (4 جمل)
- **المقدمة** (صفحة واحدة)
- المشكلة (صفحة واحدة)
- فكرتي (صفحتان)
- التفاصيل (5 صفحات)
- جهود سابقة (1-2 صفحة)
- الاستنتاج والدراسات المستقبلية (نصف صفحة)

# المقدمة (صفحة واحدة)

1. اشرح المشكلة
2. اذكر ما تقدمه لهذه المشكلة

وهذا فقط ما تكتبه....

**صفحة واحدة!**

# قم بشرح المشكلة

## 1 Introduction المقدمة

There are two basic ways to implement function application in a higher-order language, when the function is unknown: the *push/enter* model or the *eval/apply* model [11]. To illustrate the difference, consider the higher-order function **zipWith**, which zips together two lists, using a function **k** to combine corresponding list elements: **مثال للمشكلة**

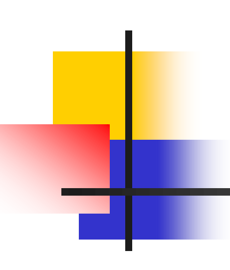
```
zipWith :: (a->b->c) -> [a] -> [b] -> [c]
zipWith k []     []     = []
zipWith k (x:xs) (y:ys) = k x y : zipWith xs ys
```

Here **k** is an *unknown function*, passed as an argument; global flow analysis aside, the compiler does not know what function **k** is bound to. How should the compiler deal with the call **k x y** in the body of **zipWith**? It can't blithely apply **k** to two arguments, because **k** might in reality take just one argument and compute for a while before returning a function that consumes the next argument; or **k** might take three arguments, so that the result of the **zipWith** is a list of functions.

**اذكر مثالا حين تقدم المشكلة للقارئ**

# اذكر مساهماتك في الموضوع

- أولا عدّد المساهمات في قائمة
- قائمة المساهمات هي التي تسوق مجرى الرسالة: وبقية الرسالة تُثَبِّت الفرضيات التي فرضتها
- سيفكر القارئ في نفسه « حقا لو انهم استطاعوا تنفيذ ذلك، فسيكون رائعا؛ علىّ أن اتابع القراءة»



# اذكر مساهماتك

Which of the two is best in practice? The trouble is that the evaluation model has a pervasive effect on the implementation, so it is too much work to implement both and pick the best. Historically, compilers for strict languages (using call-by-value) have tended to use eval/apply, while those for lazy languages (using call-by-need) have often used push/enter, but this is 90% historical accident — either approach will work in both settings. In practice, implementors choose one of the two approaches based on a qualitative assessment of the trade-offs. In this paper we put the choice on a firmer basis:

- We explain precisely what the two models are, in a common notational framework (Section 4). Surprisingly, this has not been done before.
- The choice of evaluation model affects many other design choices in subtle but pervasive ways. We identify and discuss these effects in Sections 5 and 6, and contrast them in Section 7. There are lots of nitty-gritty details here, for which we make no apology — they were far from obvious to us, and articulating these details is one of our main contributions.

In terms of its impact on compiler and run-time system complexity, eval/apply seems decisively superior, principally because push/enter requires a stack like no other: stack-walking

**المساهمات في قائمة مرقّمة**

**لا تجعل القارئ مضطرا للتخمين ما هي مساهماتك!**

# المساهمات عليها أن تكون قابلة للدحص والمناقشة

| نعم! | لا! |
|---|---|
| نشرح هنا المفردات والأوامر اللغوية المستخدمة في الأنظمة الحالية (الباب الثالث)... مميزات هذه اللغة هي ... | نشرح هنا النظام الخزعبلي فهو رائع! |
| نقوم باثبات أن نظام الطباعة سليم، وأن نظام تصحيح الأخطاء قابل للجدل (باب 4) | سندرس خواصها |
| قمنا بتحسين الجهاز الخزعبلي بوضع نظاما تفاعليا فيه يجعله أيسر للاستخدام | لقد استخدمنا النظام الخزعبلي في الواقع |

# لا تكتب «أما بقية البحث فهو...»

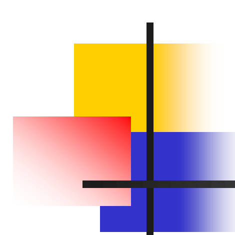
- ليس هكذا:

**بقية البحث كما يلي: الجزء الثاني مقدمة الموضوع. الجزء الثالث... وأخيرا الجزء الثامن يختم**

- الأفضل: قدّم (في مقدمة البحث) النقاط والعبارات الهامة التي تتحدث عنها في بقية البحث المقدمة (بالاضافة الى المساهمات) تعطي نظرة أولى (مسح) على البحث بأكمله. لذا قدّم (تقديم سريع) جميع الأجزاء الهامة هنا

# الهيكل

- الملخص (4 جمل)
- المقدمة (صفحة واحدة)
- ~~جهود سابقة~~
- المشكلة (صفحة واحدة)
- فكرتي (صفحتان)
- التفاصيل (5 صفحات)
- جهود سابقة (1-2 صفحة)
- الاستنتاج والدراسات المستقبلية (نصف صفحة)



# الجهود السابقة ليس الآن

الجهود السابقة

فكرتك

القارئ

We adopt the notion of transaction from Brown [1], as modified for distributed systems by White [2], using the four-phase interpolation algorithm of Green [3]. Our work differs from White in our advanced revocation protocol, which deals with the case of priority inversion as described by Yellow [4].




- السبب الأول: القارئ لا يعرف شيئا عن المشكلة حتى الآن، لذا تعدادك لملخص أعمال تتعلق بالمشكلة غير مفهوم البتة
- السبب الثاني: شرحك لحلول الآخرين يقف عائقا بين القارئ وبين فكرتك التي تريده ان يعرفها



# الهيكل

- الملخص (4 جمل)
- المقدمة (صفحة واحدة)
- المشكلة (صفحة واحدة)
- فكرتي (صفحتان)
- التفاصيل (5 صفحات)
- جهود سابقة (1-2 صفحة)
- الاستنتاج وجهود مستقبلية (نصف صفحة)

# تقديم الفكرة

**3. The idea الفكرة**

Consider a bifircuated semi-lattice D, over a hyper-modulated signature S. Suppose $p_i$ is an element of D. Then we know for every such $p_i$ there is an epi-modulus j, such that $p_j < p_i$.

- تبدو رائعة ... ولكن
- ترسل القارئ الى سبات عميق
- في أي بحث **يجب** أن تشرح التفاصيل ولكن **أولا** عليك أن توصل الفكرة للقارئ

## كيف تقدم الفكرة؟

- اشرح الفكرة وكأنك تشرحها لشخص على سبورة
- **ايصال المفهوم (وراء الفكرة) شئ أوّلي** وليس ثانوي
- بمجرد أن يصل للقارئ مفهوم الفكرة، يستطيع بعد ذلك أن يتابع التفاصيل (وليس العكس)
- وحتى لو اختارالقارئ ان يتخطى التفاصيل فانه يخرج بشئ ثمين (مفهوم الفكرة)

# ليكن القارئ أول اهتماماتك

- **لا** تقوم بسرد قصة اكتشافك للفكرة. قد يكون طريق الاكتشاف ملطخا بدمك، لكن ذلك لا يهم القارئ

- بدلا من ذلك: اختر طريقا مباشراً الى الفكرة

# الجزء المثمر في بحثك

**قدم المشكلة وفكرتك**

**بالاستعانة ب**

**أمثـلة**

**فقط بعد ذلك قدم**

**الموضوع كاملا**

# الاستعانة بالأمثلة

سؤالنا: هل يوجد خط خاص بآلات الطباعة؟

## 2 Background

To set the scene for this paper, we begin with a brief overview of the *Scrap your boilerplate* approach to generic programming. Suppose that we want to write a function that computes the size of an arbitrary data structure. The basic algorithm is "for each node, add the sizes of the children, and add 1 for the node itself". Here is the entire code for `gsize`:

```
gsize :: Data a => a -> Int
gsize t = 1 + sum (gmapQ gsize t)
```

The type for `gsize` says that it works over any type `a`, provided `a` is a *data* type — that is, that it is an instance of the class `Data`[1] The definition of `gsize` refers to the operation `gmapQ`, which is a method of the `Data` class:

```
class Typeable a => Data a where
  ...other methods of class Data...
  gmapQ :: (forall b. Data b => b -> r) -> a -> [r]
```

كن جاهزا لتقديم مثال

# التفاصيل (المتن): اثباتات

- في المقدمة وضعت ادعاءات (فرضيات)
- متن البحث يشرح اثباتات تُعزّز كل من الادعاءات
- عيّن كل من الادعاءات الموجودة في المقدمة، ثم عيّن الاثبات الذي يعنيه في التفاصيل. الآن تأكد من تقديم هذه الاثباتات في مقدمتك
- الاثباتات (الدلائل) بالامكان ان تكون: تحليلية ومقارنة، نظرية، دراسات حالة...الخ

# الهيكل

- الملخص (4 جمل)
- المقدمة (صفحة واحدة)
- المشكلة (صفحة واحدة)
- فكرتي (صفحتان)
- التفاصيل (5 صفحات)
- **جهود سابقة (1-2 صفحة)**
- الاستنتاج والجهود مستقبلية (نصف صفحة)

# جهود مسبقة

**معلومة خاطئة حتى أجعل عملي يبدو بأفضل مظهر عليّ أن أجعل أعمال الآخرين تبدو سيئة**

# حقيقة: نسبة العمل لصاحبه لا يخسرك شيئا

**عندما تنسب أي بحث تذكره لصاحبه الحقيقي لا يقلل من التقدير الذي ستلقاه من بحثك**

- اشكر الأشخاص الذين أعانوك على البحث بأسلوب دافئ (لا تكن باردا)
- كن كريما مع منافسيك: «في رسالته الملهمة < د.فلان> أثبت كذا وكذا.... ونحن نطور ما بدأه بالطرق التالية»
- كن صريحاً؛ اذكر نقاط ضعف منهجك في البحث

# حقيقة: نسبة العمل لصاحبه لا يخسرك شيئا

**تقصيرك في اعطاء التقدير للآخرين بامكانه أن يدمّر بحثك**

ادعاءك بأن فكرة ما هي فكرتك، والمقيّمين للبحث يعرفون أنها ليست فكرتك، فاما أنك:

- لا تعرف أنها فكرة قديمة (مظهرك سيّء)
- تعرف ولكنك تدّعي انها فكرتك (مظهرك سيّء جدا)

# الهيكل

- الملخص (4 جمل)
- المقدمة (صفحة واحدة)
- المشكلة (صفحة واحدة)
- فكرتي (صفحتان)
- التفاصيل (5 صفحات)
- جهود سابقة (1-2 صفحة)
- **الاستنتاج والجهود المستقبلية (نصف صفحة)**

# الاستنتاج والجهود المستقبلية

- اختصر.

# عملية الكتابة

# العملية

- ابدأ مبكراً، مبكراً جدا(لا نقصد الصباح بل مبكرا في عملية البحث)
- البحوث المكتوبة بعجلة تلقى الرفض
- البحوث كالمخللات (تحتاج وقتا لتنضج)
- تعاون
- استخدم نظام CVS للأعمال التعاونية (نظام على الانترنت يسمح بوضع الملفات عليه حتى يستطيع عدد كبير من الزملاء الاضافة فيه أو متابعته)

# الحصول على المساعدة

استعن بأكبر عدد من فئران التجارب الطيبين لقراءة بحثك

- الخبراء مفيدين
- غير الخبراء أيضا مفيدين جداً
- كل شخص يستطيع أن يقرأ بحثك (قراءة أولى) مرة واحدة فقط. فاستعن بهم بحنكة
- وضّح لهم ماذا تريد «مثلا: وجدت صعوبة هنا» أهم من سؤالك عن كيفية هجاء الكلمات

# كيف تحصل على مساعدة خبراء؟

- خطة جيدة: عندما تشعر أنك انتهيت، ارسل نسخة الى أحد منافسيك (أصحاب الأعمال السابقة) واستفسرمثلا «هل بامكانك مساعدتي في التأكد بأني شرحت عملك بالطريقة الصحيحة؟»
- غالبا ما يفيدوك بنقد مفيد (فهم مهتمين في هذا المجال)
- من الأرجح أنهم سيكونوا من المقيّمين لبحثك على كل حال. واستعانتك بنصائحهم ونقدهم جدا مفيد وفي صالحك

# استمع الى ناقديك

**تعامل مع كل نقد وكأنه ذهباً**

كن شديد الامتنان للنقد السلبي والاطراء سواء

نعرف أن هذا **جدا، جدا، جدا** صعب

ولكنه

**جدا، جدا، جدا جدا، جدا، جدا جدا، جدا، جدا جدا، جدا، جدا، جدا، جدا**

**مهم!**

# استمع الى ناقديك

- اقرأ كل نقد سلبي وكأنه اقتراحاً ايجابياً لتحسين أسلوبك أو توضيح شرحك في البحث
- لا تقم بالرد كالتالي < أيها الغبي، كنت أقصد **كَذا وكذا**> بل قم بتحسين شرحك في البحث حتى يكون **كذا وكذا** واضحا حتى لأغبى قارئ
- اشكرهم بدفء. فقد أعطوك من وقتهم الثمين

# اللغة والأسلوب

# أساسيات

- قدّم البحث في الوقت المحدّد
- ابق في اطارعدد الكلمات أو الصفحات المطلوبة (لا تزيد)

لا تغير عرض الهوامش margins

- **لا تستخدم خطّا صغيرا**
- في بعض الأحيان بامكانك وضع معلومات اضافية مفيدة للبحث (من تجارب أو أبحاث أخرى مثلا) في فهرس مخصص
- دائما استعن بمصحح الأخطاء الاملائية

# المظهر العام

- اعط بحثك مظهرا قويا فعالا باستخدام
  - sections and sub-sections فقرات وأقسام أساسية وتابعة
  - Bullets ترقيم
  - Italics خط مائل
  - في البرمجة، أكواد مسرودة بالطريقة الصحيحة
- تعلم كيفية رسم الصور واستخدمها

# المظهر العام

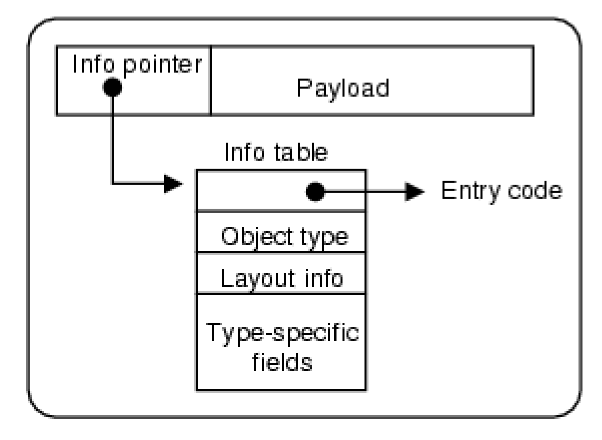



**Figure 3. A heap object**

The three cases above do not exhaust the possible forms of $f$. It might also be a *THUNK*, but we have already dealt with that case (rule THUNK). It might be a *CON*, in which case there cannot be any pending arguments on the stack, and rules UPDATE or RET apply.

## 4.3 The eval/apply model

The last block of Figure 2 shows how the eval/apply model deals with function application. The first three rules all deal with the case of a *FUN* applied to some arguments:

- If there are exactly the right number of arguments, we behave exactly like rule KNOWNCALL, by tail-calling the function. Rule EXACT is still necessary — and indeed has a direct counterpart in the implementation — because the function might not be statically known.
- If there are too many arguments, rule CALLK pushes a *call*

remainder of the object is called the *payload*, and may consist of a mixture of pointers and non-pointers. For example, the object $CON(C\ a_1 \ldots a_n)$ would be represented by an object whose info pointer represented the constructor $C$ and whose payload is the arguments $a_1 \ldots a_n$.

The info table contains:

- Executable code for the object. For example, a *FUN* object has code for the function body.
- An object-type field, which distinguishes the various kinds of objects (*FUN*, *PAP*, *CON* etc) from each other.
- Layout information for garbage collection purposes, which describes the size and layout of the payload. By "layout" we mean which fields contain pointers and which contain non-pointers, information that is essential for accurate garbage collection.
- Type-specific information, which varies depending on the object type. For example, a *FUN* object contains its arity; a *CON* object contains its constructor tag, a small integer that distinguishes the different constructors of a data type; and so on.

In the case of a PAP, the size of the object is not fixed by its info table; instead, its size is stored in the object itself. The layout of its fields (e.g. which are pointers) is described by the (initial segment of) an argument-descriptor field in the info table of the FUN object which is always the first field of a PAP. The other kinds of heap object all have a size that is statically fixed by their info table.

A very common operation is to jump to the entry code for the object, so GHC uses a slightly-optimised version of the representation in Figure 3. GHC places the info table at the addresses *immediately*

# ابتعد عن الأسلوب المبني للمجهول

**الاسلوب المبني للمجهول أي نعم علمياً،**

**ولكنه يميت البحث. فتفاداه بكل الطرق**

| نعم | لا |
| --- | --- |
| بامكاننا أن نشاهد.... | يُشاهد أن.... |
| أجرينا 43 تجربة | تم اجراء 43 تجربة |
| أردنا المحافظة على هذه الخواص | عُدّت هذه الخواص هامّة |
| قد تعتقد أن هذا خطأ مطبعياً | قد يُعتقد بأن ذلك خطأ مطبعياّ |

بامكاننا (نحن): أنت والقارئ

أردنا (نحن) المؤلفين

أنت أيها القارئ

ملحوظة من المترجمة: النمط في الغرب الآن الابتعاد عن المبني للمجهول حتى في الكتابات العلمية الا في مواضع معينة ، تأكد من رأي منشأتك ومن في مجالك في هذه النقطة

# استخدم لغة مبسطة ومحددة

| نعم | لا |
| --- | --- |
| تحركت الكرة بانحراف | المتغير تحت الدراسة تم تحريكه في مسار منحرف |
| سنوياً | بمعدل سنوي |
| لنعرف | لنقوم باستنباط المعلومة |
| عامل النظافة كان بطيئاً | نستطيع أن نقول بأن عملية جمع النفايات لم تكن على الوجه المتوقع |

# الملخص

أهم ما عليك تذكره

- عرّف القارئ بفكرتك الأساسية
- اجعل مساهماتك واضحة
- استعن بالأمثلة

مرجع مفيد للمبتدئين

"Advice on Research and Writing"

http://www-2.cs.cmu.edu/afs/cs.cmu.edu/user/mleone/web/how-to.html